Project Assignment

Project Title: Building a Robust MLOps Pipeline

Course: ID5003W
Academic Year: 2025
Submission Deadline : 29th August, 2025

Department of Data Science and Artificial Intelligence
IIT Madras

Chapter 1

Project Overview

1.1 Introduction and Motivation

In the modern era of data science, building a machine learning model with high accuracy is only the first step. The true challenge lies in deploying, scaling, managing, and maintaining these models in production environments where data evolves and business needs change. This operational discipline, known as MLOps (Machine Learning Operations), is critical for creating real-world value from AI.

This project is designed to provide you with hands-on experience in building an end-to-end, production-grade MLOps pipeline. You will move beyond the theoretical aspects of model building and tackle the practical engineering challenges of creating a system that is scalable, automated, reproducible, and robust.

1.2 Project Goal

The primary goal of this project is to design, implement, and document a comprehensive MLOps pipeline. You will leverage standard tools for big data processing, experiment tracking, data versioning, and model serving. The final output should be a system that can automatically train, evaluate, deploy, and monitor machine learning models at scale. The project will be judged not only on the performance of the final model but, more importantly, on the quality, robustness, and automation of the pipeline you create.

Chapter 2

Problem Statement and Scope

2.1 Core Task

You are tasked with building a complete MLOps pipeline to train, deploy, and maintain a classification model. The pipeline must be able to handle large datasets efficiently and adapt to new data over time. You will need to address the entire model lifecycle, from data ingestion to production monitoring.

2.2 Datasets

You are required to demonstrate your pipeline's capabilities using one of the following two standard datasets.

1. **Kaggle's Titanic Dataset:** A classic dataset for binary classification. Your task is to predict the survival of passengers. This will be the primary dataset for building the core pipeline for structured data.

2. **MNIST Dataset:** A database of handwritten digits. Your task is to build an image classifier. For this dataset, you are required to explore and implement a transfer learning approach.

Chapter 3

Detailed Requirements and Tasks

You must address all the following requirements in your project. Your implementation should be modular, and each component should be clearly documented.

3.1 Task 1: Data Pipeline and Versioning

- **Distributed Preprocessing:** Implement a data preprocessing and feature engineering pipeline using **Apache Spark**. This pipeline should be scalable and handle all necessary transformations for both datasets.
- **Data Versioning:** Use **Data Version Control (DVC)** to version your datasets. Your project repository (Git) should not contain the raw data files but rather the DVC pointers, ensuring reproducibility.

3.2 Task 2: Model Development and Training

- **Distributed Training:** Train your models using Spark's distributed capabilities.
  - Use **Spark MLlib / BigDL** (with Hyperparameter tuning, if applicable) to train your model.
- **Automated Hyperparameter Tuning:** You can explore tools **AutoML** techniques to train your model.

`NOTE: Kindly use any one of the strategies. Do not mix both.`

3.3 Task 3: MLOps - Experiment and Model Management

- **Experiment Tracking:** Integrate **MLflow Tracking** into your training scripts. For every experiment, you must log parameters, performance metrics, and relevant artifacts (e.g., confusion matrix, feature importance plots).

- **Model Registry:** Use the **MLflow Model Registry** to manage your trained models. Your pipeline should be able to automatically register the best performing model and transition it between stages (e.g., from Staging to Production).

3.4 Task 4: Model Deployment and Testing

- **API Deployment:** Package your best model (from the MLflow Registry) and expose it as a **REST API**. You can use a framework like Flask or FastAPI. The API should accept new data and return predictions.
- **Testing:** Write a script to test the deployed API endpoint with a sample from your test dataset to verify its functionality.

3.5 Task 5: Advanced Topics — Future-Proofing and Robustness

- **Handling Distributional Shift:** Design and implement a mechanism to detect distributional shifts in new incoming data. If a significant drift is detected, your system should be able to trigger an alert or an automated retraining process.
- **Automated Retraining:** Your pipeline should be designed to support future-proofing. Create a script or workflow that simulates the arrival of new data and automatically triggers the entire pipeline to retrain, evaluate, and potentially deploy a new model.
- **Resource Optimization:** Analyze the resource consumption (e.g., core usage, memory) of your Spark training jobs. Find a near-optimal training scheme and justify your choice in the report.

`NOTE`: You can manipulate the data as per requirement. All ideas are welcome, but kindly mention it in your report.

Chapter 4

Deliverables

Your final submission must include the following components.

4.1 Project Report

You must submit a comprehensive project report in PDF format, written in LaTeX/Doc. The report should be well-structured and professionally written. A suggested structure is:

1. Abstract
2. Introduction (Problem, Motivation, Objectives)
3. System Architecture and Design (Include a high-level diagram of your pipeline)
4. Tools and Resources used in the project with proper justification
5. Implementation Details (Detail your work on each task)
6. Results and Analysis (Model performance tables, resource usage graphs, API test results)
7. Conclusion and Future Work
8. Bibliography/References

4.2 Code Repository

A link to a Git repository (e.g., on GitHub/GitLab) containing all your source code, DVC files, Dockerfile for the API, and a `README.md` file with clear instructions on how to set up and run your project.

4.3 Project Demo

Record a 10 minute video explaining and running each step of your pipeline.

Chapter 5

Evaluation Criteria

The project will be evaluated based on the following criteria. The emphasis is on the end-to-end pipeline quality over just the final model's accuracy.

Pipeline Quality:

- Correct and efficient use of Spark for data processing.
- Selection of Models used for training
- Proper integration and use of MLflow and DVC.
- Degree of automation achieved (especially for retraining and deployment).
- Robustness of the pipeline design and implementation of drift detection.

Model Performance & Analysis:

- Correct implementation of models and transfer learning.
- Thoroughness of experiments and hyperparameter tuning.
- Quality of analysis of results and resource usage.

Report and Code Quality:

- Clarity, structure, and completeness of the final report.
- Code readability, modularity, and documentation (comments, README).

Presentation and Demo:

- Clarity of the presentation and effectiveness of the demonstration.

— All the Best! —